Theoretical Computer Science 412 (2011) 3731–3737

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Semi-online scheduling with known partial information about job sizes on two identical machines

Qian Cao [a], Zhaohui Liu [b], T.C.E. Cheng [c,*]

[a] Antai College of Economics and Management, Shanghai Jiao Tong University, Shanghai, China

[b] Department of Mathematics, East China University of Science and Technology, Shanghai, China

[c] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 2 November 2010
Received in revised form 28 February 2011
Accepted 29 March 2011
Communicated by D.-Z. Du

Keywords:
Scheduling
Identical machines
Semi-online
Maximum job

ABSTRACT

In this paper we consider the semi-online scheduling problem with known partial information about job sizes on two identical machines, where all the jobs have processing times in the interval $[p, tp]$ $(p > 0, t \geq 1)$ and the maximum job size is tp. The objective is to minimize the makespan. For $1 \leq t < \frac{4}{3}$ and $t \geq 2$, we obtain lower bounds $\frac{t+1}{2}$ and $\frac{4}{3}$ on the optimal solution, respectively, which match the upper bounds given by He and Zhang (1999) in [2]. For $\frac{4}{3} \leq t < 2$, we prove that a lower bound on the optimal solution is $\max\{\frac{4t+4}{3t+4}, \frac{2t}{t+1}\}$ and design an algorithm with a competitive ratio equal to this lower bound.

1. Introduction

In the classical online scheduling problem, it is assumed that jobs arrive one by one and the current job must be scheduled irrevocably before the next job emerges. In contrast, in the offline version, we have full information about the jobs before they arrive. However, practical scheduling problems are between these two extreme cases. Such problems are known as the semi-online scheduling problem, where partial information about future jobs is available.

In this paper we consider the semi-online scheduling problem with known partial information about job sizes, i.e., we know in advance that all the jobs have sizes in the interval $[p, tp]$ $(p > 0, t \geq 1)$ and the maximum job size $p_{\max}$ is tp. In fact, this problem is a combination of the semi-online scheduling problems with bounded job sizes and with known maximum job size. For the problem under study, a list $L = (J_1, J_2, \ldots, J_n)$ of n jobs that are to be assigned to two identical machines M_1 and M_2 is given. Each job J_j is associated with a size p_j. For notational convenience, we also use p_j to represent job J_j. Our goal is to construct a schedule that minimizes the makespan, $C_{\max}$, i.e., the maximum of the job completion times on M_1 and M_2. Without loss of generality, we suppose that $p = 1$ and denote the problem by $P2|1 \leq p_j \leq p_{\max} = t|C_{\max}$.

As for the online version, we measure the performance of a semi-online algorithm $\mathcal{H}$ by its competitive ratio with respect to an optimal offline algorithm. Let $C_{\max}^{\mathcal{H}}$ denote the makespan of the schedule produced by a semi-online algorithm and $C_{\max}^{*}$ denote the optimal makespan of an offline schedule. Then the competitive ratio of algorithm $\mathcal{H}$ is defined as

$$r_{\mathcal{H}} = \inf_r \{r \geq 1 |\, C_{\max}^{\mathcal{H}} \leq r C_{\max}^{*}\}.$$

We call c a lower bound on the optimal solution for the problem if there is no semi-online algorithm with a competitive ratio less than c. Accordingly, algorithm $\mathcal{H}$ is called optimal if its competitive ratio is equal to some lower bound.

* Corresponding author. Tel.: +852 2766 5216; fax: +852 2364 5245.
E-mail addresses: caoqiansh@gmail.com (Q. Cao), zhliu@ecust.edu.cn (Z. Liu), lgtcheng@polyu.edu.hk (T.C.E. Cheng).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.03.032

To treat the online version of the scheduling problem under study, we can use a simple algorithm – the list scheduling (*LS*) algorithm – that assigns the current job to the machine with a smaller current workload. Graham [3] first considers using the list scheduling algorithm to solve scheduling problems. Faigle et al. [4] prove that the *LS* algorithm has a competitive ratio $\frac{3}{2}$ and is optimal. This provides an upper bound for the semi-online scheduling problem on two identical machines to minimize the makespan. Furthermore, He and Zhang [2] study two semi-online scheduling problems on two identical machines. The first one is a semi-online problem with bounded job sizes, in which all the jobs have processing times between p and tp ($p > 0, t \geq 1$), but it is possible that no jobs with sizes p and tp come up. They show that the *LS* algorithm has a competitive ratio $\min\{\frac{t+1}{2}, \frac{3}{2}\}$, which provides an upper bound for our problem. The second one is a semi-online problem with known maximum job size, where the maximum size of all the jobs $p_{\max}$ is known in advance. They propose an optimal algorithm *PLS* with a competitive ratio $\frac{4}{3}$, which provides another upper bound for our problem.

Researchers have studied different cases of the semi-online scheduling problem with partial information about job sizes on two identical machines to minimize the makespan. Kellerer et al. [7] consider the case where the total size of all the jobs is known in advance. They design an optimal algorithm with a competitive ratio $\frac{4}{3}$. Seiden et al. [8] study the case where the jobs arrive in decreasing order of sizes. They prove that the *LS* algorithm is optimal and has a competitive ratio $\frac{7}{6}$. Tan and He [1] consider the case where the maximum job size is between p and tp ($p > 0, t \geq 1$). They present an algorithm with a competitive ratio $\frac{2t+2}{t+2}$ for $1 \leq t \leq 2$. Tan et al. [5] consider two cases of combined semi-online scheduling problems on two identical machines. One case is where the total size of all the jobs is known in advance and the jobs arrive in decreasing order of sizes. The other case is where both the total size of all the jobs and the maximum job size are known in advance. They give optimal algorithms for the two cases with competitive ratios $\frac{10}{9}$ and $\frac{6}{5}$, respectively. Epstein [6] considers the case with combined information, where the optimal solution value is known and the jobs arrive in decreasing order of sizes, and provides an optimal algorithm with a competitive ratio $\frac{10}{9}$.

This paper is organized as follows: In Section 2 we obtain the following lower bounds on the optimal solution of the problem under study

$$\begin{cases} \dfrac{t+1}{2}, & 1 \leq t < \dfrac{4}{3} \\ \dfrac{4t+4}{3t+4}, & \dfrac{4}{3} \leq t < \sqrt{2} \\ \dfrac{2t}{t+1}, & \sqrt{2} \leq t < 2 \\ \dfrac{4}{3}, & t \geq 2. \end{cases}$$

For $1 \leq t < \frac{4}{3}$ and $t \geq 2$, the respective lower bounds $\frac{t+1}{2}$ and $\frac{4}{3}$ match the upper bounds given in [2]. In Section 3, for $1 \leq t \leq 2$, we design an algorithm *PIJS* with a competitive ratio $\max\{\frac{4t+4}{3t+4}, \frac{2t}{t+1}\}$, which is optimal for $\frac{4}{3} \leq t \leq 2$.

2. Lower bounds

Theorem 1. *For the problem $P2|1 \leq p_j \leq p_{\max} = t|C_{\max}$, the competitive ratio of an arbitrary semi-online algorithm is not less than*

$$\min\left\{\frac{2t}{t+1}, \frac{4}{3}\right\} = \begin{cases} \dfrac{2t}{t+1}, & 1 \leq t < 2 \\ \dfrac{4}{3}, & t \geq 2. \end{cases}$$

Proof. Case 1. $1 \leq t < 2$.

Let $p_1 = t$ and $p_2 = 1$. If both p_1 and p_2 are assigned to the same machine, we have $C_{\max} = t + 1$ and $C^*_{\max} = t$. Therefore

$$\frac{C_{\max}}{C^*_{\max}} = \frac{t+1}{t} \geq \frac{2t}{t+1},$$

where the last inequality holds for $t < 2 < 1 + \sqrt{2}$. Next we only need to consider the case where p_1 and p_2 are assigned to different machines. Without loss of generality, assume that p_1 is assigned to M_1 and p_2 is assigned to M_2. A new job $p_3 = 1$ arrives. If p_3 is assigned to M_1, then no other jobs arrive. Since $C_{\max} = t + 1$ and $C^*_{\max} = 2$, we have

$$\frac{C_{\max}}{C^*_{\max}} = \frac{t+1}{2} \geq \frac{2t}{t+1},$$

where the last inequality holds for $t \geq 1$. If p_3 is assigned to M_2, then the last job $p_4 = t$ arrives. Then $C_{\max} \geq \min\{2t, t+2\} = 2t$ and $C^*_{\max} = t + 1$. Therefore

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{2t}{t+1}.$$

Case 2. $t \geq 2$.

Let $p_1 = t$ and $p_2 = \frac{t}{2} \geq 1$. If both p_1 and p_2 are assigned to the same machine, we have $C_{\max} = t + \frac{t}{2} = \frac{3t}{2}$ and $C^*_{\max} = t$. Therefore,

$$\frac{C_{\max}}{C^*_{\max}} = \frac{3}{2} > \frac{4}{3}.$$

As in the proof of Case 1, we assume that p_1 is assigned to M_1 and p_2 is assigned to M_2. A new job $p_3 = \frac{t}{2}$ arrives. If p_3 is assigned to M_1, then no other jobs arrive. Since $C_{\max} = \frac{3t}{2}$ and $C^*_{\max} = t$, it holds that

$$\frac{C_{\max}}{C^*_{\max}} = \frac{3}{2} > \frac{4}{3}.$$

If p_3 is assigned to M_2, the last job $p_4 = t$ arrives. Then $C_{\max} = 2t$ and $C^*_{\max} = t + \frac{t}{2} = \frac{3t}{2}$. Therefore

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{4}{3}. \quad \square$$

Theorem 2. *For the problem $P2|1 \leq p_j \leq p_{\max} = t|C_{\max}$, the competitive ratio of an arbitrary semi-online algorithm is not less than $\frac{t+1}{2}$ for $1 \leq t \leq \frac{4}{3}$.*

Proof. Let $p_1 = p_2 = t$. If both p_1 and p_2 are assigned to the same machine, we have $C_{\max} = 2t$ and $C^*_{\max} = t$. Therefore

$$\frac{C_{\max}}{C^*_{\max}} = 2 \geq \frac{t+1}{2}.$$

Next we only need to consider the case where p_1 and p_2 are assigned to different machines.

Four new jobs $p_3 = p_4 = p_5 = p_6 = 1$ arrive. If at least three of the jobs in $\{p_3, p_4, p_5, p_6\}$ are assigned to the same machine, we have $C_{\max} \geq t + 3$ and $C^*_{\max} = t + 2$. Therefore,

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{t+3}{t+2} \geq \frac{t+1}{2},$$

where the last inequality holds for $t \leq \frac{4}{3} < \frac{-1+\sqrt{17}}{2}$.

Otherwise, the last job $p_7 = t$ arrives. Since $C_{\max} = 2t + 2$ and

$$C^*_{\max} \leq \max\{4, 3t\} = 4,$$

it holds that

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{t+1}{2}. \quad \square$$

Theorem 3. *For the problem $P2|1 \leq p_j \leq p_{\max} = t|C_{\max}$, the competitive ratio of an arbitrary semi-online algorithm is not less than $\frac{4t+4}{3t+4}$ for $\frac{4}{3} \leq t \leq \sqrt{2}$.*

Proof. Let $p_1 = t$ and $p_2 = 1$. If both p_1 and p_2 are assigned to the same machine, we have $C_{\max} = t + 1$ and $C^*_{\max} = t$. Therefore,

$$\frac{C_{\max}}{C^*_{\max}} = \frac{t+1}{t} \geq \frac{4t+4}{3t+4},$$

where the last inequality holds for $t \leq \sqrt{2} < 4$. Without loss of generality, we assume that p_1 is assigned to M_1 and p_2 is assigned to M_2 in the following.

A new job $p_3 = \frac{3t}{2} - 1$ arrives. Note that $1 \leq p_3 \leq t$ for $\frac{4}{3} \leq t \leq 2$. If p_3 is assigned to M_1, then no other jobs arrive. Since $C_{\max} \geq p_1 + p_3 = \frac{5t}{2} - 1$ and $C^*_{\max} \leq \max\{p_1, p_2 + p_3\} = \frac{3t}{2}$, we have

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{5t-2}{3t} \geq \frac{4t+4}{3t+4},$$

where the last inequality holds for $t \geq \frac{4}{3}$.

If p_3 is assigned to M_2, then a job $p_4 = 1$ arrives. If p_4 is assigned to M_2, then no other jobs arrive. Hence $C_{\max} \geq p_2 + p_3 + p_4 = \frac{3t}{2} + 1$, but

$$C^*_{\max} \leq \max\{p_1 + p_4,\ p_2 + p_3\} = \max\left\{t + 1,\ \frac{3t}{2}\right\} = t + 1,$$

where the last inequality holds for $t \leq \sqrt{2} < 2$. Therefore

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{3t+2}{2t+2} \geq \frac{4t+4}{3t+4},$$

where the last inequality holds for an arbitrary t.

If p_4 is assigned to M_1, then a job $p_5 = 2 - \frac{t}{2}$ arrives. Note that $1 \leq p_5 \leq t$ for $\frac{4}{3} \leq t \leq 2$. If p_5 is assigned to M_1, then no other jobs arrive. Hence $C_{\max} \geq p_1 + p_4 + p_5 = \frac{t}{2} + 3$, but

$$C^*_{\max} \leq \max\{p_1 + p_5, p_2 + p_3 + p_4\} = \max\left\{\frac{t}{2} + 2,\ \frac{3t}{2} + 1\right\} = \frac{3t}{2} + 1.$$

Therefore,

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{t+6}{3t+2} \geq \frac{4t+4}{3t+4},$$

where the last inequality holds for $t \leq \sqrt{2} < \frac{1+\sqrt{145}}{9} \simeq 1.449$.

If p_5 is assigned to M_2, then a job $p_6 = 1$ arrives. If p_6 is assigned to M_2, then no other jobs arrive. Since $C_{\max} \geq p_2 + p_3 + p_5 + p_6 = t + 3$ and

$$C^*_{\max} \leq \max\{p_1 + p_4 + p_6, p_2 + p_3 + p_5\} = p_1 + p_4 + p_6 = p_2 + p_3 + p_5 = t + 2,$$

it holds that

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{t+3}{t+2} \geq \frac{4t+4}{3t+4},$$

where the last inequality holds for $t \leq \sqrt{2} < \frac{1+\sqrt{17}}{2} \simeq 2.562$.

If p_6 is assigned to M_1, then the last job $p_7 = t$ arrives. No matter how we assign job p_7, it holds that

$$C_{\max} = p_2 + p_3 + p_5 + p_7 = p_1 + p_4 + p_6 + p_7 = 2t + 2$$

and

$$C^*_{\max} = p_2 + p_3 + p_4 + p_6 = p_1 + p_5 + p_7 = \frac{3t}{2} + 2.$$

Therefore,

$$\frac{C_{\max}}{C^*_{\max}} \geq \frac{2t+2}{\frac{3t}{2}+2} = \frac{4t+4}{3t+4}. \quad \square$$

In summary, we obtain the following lower bounds

$$\begin{cases} \dfrac{t+1}{2}, & 1 \leq t < \dfrac{4}{3} \\ \dfrac{4t+4}{3t+4}, & \dfrac{4}{3} \leq t < \sqrt{2} \\ \dfrac{2t}{t+1}, & \sqrt{2} \leq t < 2 \\ \dfrac{4}{3}, & t \geq 2 \end{cases}$$

by Theorems 1–3. As mentioned in Section 1, He and Zhang [2] give an upper bound $\min\{\frac{t+1}{2}, \frac{4}{3}\}$ for our problem, so we only need to use the *LS* algorithm with a competitive ratio $\frac{t+1}{2}$ for $1 \leq t \leq \frac{4}{3}$ to match our lower bound in Theorem 2 and the *PLS* algorithm with a competitive ratio $\frac{4}{3}$ for $t \geq 2$ to match our lower bound in Theorem 1. In the next section we solve the case where $\frac{4}{3} \leq t \leq 2$, which completes the analysis of the problem.

3. Algorithm

In this section, for $1 \leq t \leq 2$, we design an algorithm *PIJS* with a competitive ratio $r_{PIJS} = \max\{\frac{4t+4}{3t+4}, \frac{2t}{t+1}\}$, which is optimal for $\frac{4}{3} \leq t \leq 2$ by Theorems 1 and 3. For the problem $P2|1 \leq p_j \leq p_{\max} = t|C_{\max}$, we know that a job of size $p_{\max} = t$ will arrive, so it is possible to schedule such a job in advance. Applying this idea, we schedule a job of size $p_{\max}$ in advance on M_2 and the remaining jobs on M_1 unless the workload of M_1 exceeds $r_{PIJS}C^*_{\max}$. Since $C^*_{\max}$ is unknown, we replace it by its lower bound. Next we introduce some notation. We denote the workload of M_i $(i = 1, 2)$ before the assignment of p_j $(j = 1, 2, \ldots, n)$ by M_i^j. Let $q_i^{(j)}$ be the ith smallest job when p_j appears, i.e., $\{q_1^{(j)}, q_2^{(j)}, \ldots, q_j^{(j)}\} = \{p_1, p_2, \ldots, p_j\}$, where $q_1^{(j)} \leq q_2^{(j)} \leq \cdots \leq q_j^{(j)}$ and $j = 1, 2, \ldots, n$. Because there are at least $\lceil \frac{n}{2} \rceil$ jobs that are assigned to one of the two machines by an optimal offline algorithm, we have $C^*_{\max} \geq q_1^{(n)} + q_2^{(n)} + \cdots + q^{(n)}_{\lceil \frac{n}{2} \rceil}$. We also use M_i to denote the final workload of machine M_i, so $C^*_{\max} \geq \frac{M_1+M_2}{2}$. Let $C^*_{\max}(j)$ be the optimal makespan of an offline schedule for $p_1, p_2, \ldots, p_j$. Therefore, if p_j appears before the first $p_{\max}$, we have

$$C^*_{\max}(j) \geq \max\left\{q_1^{(j)} + q_2^{(j)} + \cdots + q^{(j)}_{\lceil \frac{j+1}{2} \rceil}, \frac{M_1^j + M_2^j + p_j + p_{\max}}{2}\right\};$$

otherwise

$$C^*_{\max}(j) \geq \max\left\{q_1^{(j)} + q_2^{(j)} + \cdots + q^{(j)}_{\lceil \frac{j}{2} \rceil}, \frac{M_1^j + M_2^j + p_j}{2}\right\}.$$

Algorithm *PIJS*

Step 1. Schedule the current job p_j using the following rule:

if $\quad M_1^j + p_j \leq r_{PIJS} \cdot \max\left\{q_1^{(j)} + q_2^{(j)} + \cdots + q^{(j)}_{\lceil \frac{j+1}{2} \rceil}, \frac{M_1^j+M_2^j+p_j+p_{\max}}{2}\right\}$

then $\quad p_j \to M_1$

else $\quad p_j \to M_2$,

until the first $p_{\max}$ emerges.

Step 2. Schedule the first $p_{\max}$ on M_2.

Step 3. Schedule the remaining jobs using the following rule:

if $\quad M_1^j + p_j \leq r_{PIJS} \cdot \max\left\{q_1^{(j)} + q_2^{(j)} + \cdots + q^{(j)}_{\lceil \frac{j}{2} \rceil}, \frac{M_1^j+M_2^j+p_j}{2}\right\}$

then $\quad p_j \to M_1$

else $\quad p_j \to M_2$.

Note that $r_{PIJS} = r = \max\{\frac{4t+4}{3t+4}, \frac{2t}{t+1}\}$ in algorithm *PIJS*. In fact, we can assume that p_1 is the first largest job $p_{\max}$. Otherwise, let p_m $(m = 2, 3, \ldots, n)$ be the first largest job $p_{\max}$. We obtain a new job list $L' = (p_m, p_1, \ldots, p_{m-1}, p_{m+1}, \ldots, p_n)$ by transferring job p_m to the first position while leaving all the other jobs in their original positions. Under the new job list L', the assignment of each job to a machine is not changed by algorithm *PIJS*.

Since $C^*_{\max} \geq p_{\max}$, algorithm *PIJS* is optimal for $n = 1$ and $n = 2$. (For $n = 2$, the first largest job $p_1 = p_{\max}$ is assigned to M_2 and the other job p_2 is assigned to M_1 by algorithm *PIJS*.) Next we only need to consider the case where $n \geq 3$. Without loss of generality, we suppose that $C^{PIJS}_{\max}$ is determined by p_n, i.e., p_n is the last finished job.

Lemma 1. *If p_n is assigned to M_1, then $\frac{C^{PIJS}_{\max}}{C^*_{\max}} \leq r$.*

Proof. Since p_n is not the first largest job $p_{\max}$ and it is assigned by Step 3 of algorithm *PIJS*, we have

$$C^{PIJS}_{\max} = M_1 = M_1^n + p_n \leq r \cdot \max\left\{q_1^{(n)} + q_2^{(n)} + \cdots + q^{(n)}_{\lceil \frac{n}{2} \rceil}, \frac{M_1 + M_2}{2}\right\} \leq rC^*_{\max}. \quad \square$$

Lemma 2. *If p_n is assigned to M_2 and at least $\lceil \frac{n}{2} \rceil$ jobs are assigned to M_1, then $\frac{C^{PIJS}_{\max}}{C^*_{\max}} \leq r$.*

Proof. Since $1 \leq p_j \leq t$ for $1 \leq j \leq n$, it holds that

$$\frac{M_1}{M_2} \geq \frac{\lceil \frac{n}{2} \rceil}{(n - \lceil \frac{n}{2} \rceil)t} \geq \frac{1}{t}.$$

Therefore,

$$\frac{C^{PIJS}_{\max}}{C^*_{\max}} = \frac{M_2}{C^*_{\max}} \leq \frac{M_2}{\frac{M_1+M_2}{2}} = \frac{2}{\frac{M_1}{M_2} + 1} \leq \frac{2}{\frac{1}{t} + 1} = \frac{2t}{t+1} \leq r. \quad \square$$

Corollary 1. *For $n = 3$, if p_n is assigned to M_2, then $\frac{C^{PIJS}_{\max}}{C^*_{\max}} \le r$.*

Proof. It is clear that job $p_1 = p_{\max}$ is assigned to M_2. Since $p_2 \le \frac{p_1+p_2}{2} \le r \cdot \frac{p_1+p_2}{2}$ and $p_2+p_3 = q_1^{(3)} + q_2^{(3)} \le r \cdot (q_1^{(3)} + q_2^{(3)})$, algorithm *PIJS* assigns both jobs p_2 and p_3 to M_1. So the conclusion holds by Lemma 2. □

Lemma 3. *If p_n is assigned to M_2 and at most $\lceil \frac{n}{2} \rceil - 1$ jobs are assigned to M_1, then $\frac{C^{PIJS}_{\max}}{C^*_{\max}} \le r$ for $1 \le t \le 2$.*

Proof. It is clear that at least $n - (\lceil \frac{n}{2} \rceil - 1) = n + 1 - \lceil \frac{n}{2} \rceil$ jobs are assigned to M_2. We perform a case-by-case analysis in the following.

Case 1. $n = 4$.

We know that at least three jobs are assigned to M_2 and at most one job is assigned to M_1. Note that p_n is not the first largest job $p_{\max}$, so it holds that $M_1^n + p_n \le M_1 + p_n \le 2t \le t(p_{\max} + 1) \le tM_2^n$. Thus

$$M_1^n + p_n \le \frac{2t}{t+1} \cdot \frac{M_1^n + M_2^n + p_n}{2} \le r \cdot \frac{M_1^n + M_2^n + p_n}{2}$$

and p_n is assigned to M_1 by algorithm *PIJS*, a contradiction.

Case 2. $n = 5$.

Note that $C^*_{\max} \ge q_1^{(5)} + q_2^{(5)} + q_3^{(5)}$. If at most one job is assigned to M_1 by algorithm *PIJS*, then $M_1 + p_n \le 2t \le \frac{6t}{t+1} \le \frac{2t}{t+1}(q_1^{(5)} + q_2^{(5)} + q_3^{(5)}) \le r(q_1^{(5)} + q_2^{(5)} + q_3^{(5)})$. So p_n is assigned to M_1 by algorithm *PIJS*, a contradiction. We only need to consider the case where two jobs are assigned to M_1 and three jobs are assigned to M_2 by algorithm *PIJS*. If there are at least two common jobs between the jobs assigned to M_2 and $\{q_1^{(5)}, q_2^{(5)}, q_3^{(5)}\}$, then

$$\frac{C^{PIJS}_{\max}}{C^*_{\max}} \le \frac{M_2}{q_1^{(5)} + q_2^{(5)} + q_3^{(5)}} \le 1 + \frac{t-1}{3} = \frac{t+2}{3} \le \frac{2t}{t+1} \le r,$$

where the second last inequality holds for $1 \le t \le 2$. Otherwise,

$$\frac{M_1 + p_n}{q_1^{(5)} + q_2^{(5)} + q_3^{(5)}} \le 1 + \frac{t-1}{3} = \frac{t+2}{3} \le \frac{2t}{t+1} \le r.$$

So p_n is assigned to M_1 by algorithm *PIJS*, a contradiction.

Case 3. $n = 6$.

We know that at least four jobs are assigned to M_2 and at most two jobs are assigned to M_1. Note that p_n is not the first largest job $p_{\max}$, so it holds that $M_1^n + p_n \le M_1 + p_n \le 3t \le t(p_{\max} + 2) \le tM_2^n$. Thus

$$M_1^n + p_n \le \frac{2t}{t+1} \cdot \frac{M_1^n + M_2^n + p_n}{2} \le r \cdot \frac{M_1^n + M_2^n + p_n}{2}$$

and p_n is assigned to M_1 by algorithm *PIJS*, a contradiction.

Case 4. $n \ge 7$.

Note that there must be at least four jobs assigned to M_2. Since p_n is not the first largest job $p_{\max}$ and it is assigned by Step 3 of algorithm *PIJS*, we have

$$M_1^n + p_n > r \cdot \frac{M_1^n + M_2^n + p_n}{2}.$$

Then $M_1 + p_n \ge M_1^n + p_n > \frac{r}{2-r} \cdot M_2^n = \frac{r}{2-r}(M_2 - p_n)$. Hence

$$\frac{M_1}{M_2} \ge \frac{r}{2-r} - \left(\frac{r}{2-r} + 1\right)\frac{p_n}{M_2} \ge \frac{r}{2-r} - \left(\frac{r}{2-r} + 1\right)\frac{p_n}{t+2+p_n} \ge \frac{r}{2-r} - \left(\frac{r}{2-r} + 1\right)\frac{t}{2t+2}.$$

Therefore, we have

$$\frac{C^{PIJS}_{\max}}{C^*_{\max}} \le \frac{M_2}{\frac{M_1+M_2}{2}} = \frac{2}{\frac{M_1}{M_2} + 1} \le \frac{2}{\frac{r}{2-r} - (\frac{r}{2-r} + 1)\frac{t}{2t+2} + 1}$$
$$= \frac{2}{(1 - \frac{t}{2t+2})(\frac{r}{2-r} + 1)} = \frac{2}{\frac{t+2}{2t+2} \cdot \frac{2}{2-r}} = \frac{2(t+1)(2-r)}{t+2} \le r,$$

where the last inequality holds for $r \ge \frac{4t+4}{3t+4}$. □

Theorem 4. *For* $1 \le t \le 2$, *we have*

$$\frac{C_{\max}^{PIJS}}{C_{\max}^{*}} \le r = \max\left\{\frac{4t+4}{3t+4}, \frac{2t}{t+1}\right\} = \begin{cases} \dfrac{4t+4}{3t+4}, & 1 \le t < \sqrt{2} \\ \dfrac{2t}{t+1}, & \sqrt{2} \le t \le 2. \end{cases}$$

Proof. See Lemmas 1–3. □

Acknowledgements

We are grateful to an anonymous referee for his/her thoughtful comments and suggestions on an earlier version of our paper. Cheng was supported in part by The Hong Kong Polytechnic University under grant number G-U720.

References

[1] Z.Y. Tan, Y. He, Semi-online scheduling problems on two identical machines with inexact partial information, Theoretical Computer Science 377 (2007) 110–125.
[2] Y. He, G. Zhang, Semi on-line scheduling on two identical machines, Computing 62 (1999) 179–187.
[3] R.L. Graham, Bounds for certain multiprocessor anomalies, Bell Systems Technical Journal 45 (1966) 1563–1581.
[4] U. Faigle, W. Kern, G. Turan, On the performance of online algorithms for partition problems, Acta Cybernetica 9 (1989) 107–119.
[5] Z.Y. Tan, Y. He, Semi-online problems on two identical machines with combined partial information, Operations Research Letters 30 (2002) 408–414.
[6] L. Epstein, Bin stretching revisited, Acta Informatica 39 (2003) 97–117.
[7] H. Kellerer, V. Kotov, M.G. Speranza, Z. Tuza, Semi on-line algorithms for the partition problem, Operations Research Letters 21 (1997) 235–242.
[8] S. Seiden, J. Sgall, G. Woeginger, Semi-online scheduling with decreasing job sizes, Operations Research Letters 27 (2000) 215–227.